Skaffold introduction –{local using}

Agenda

1. Definition
2. Pre-requisite
3. Skaffold code basic
4. Workflow
5. Core feature
6. How to use
7. Using scenarios
8. Conclusion
9. Feedback

Definition

- Skaffold is a command line tool that facilitates continuous development for Kubernetes-native applications.
- Skaffold handles the workflow for **building**, **pushing**, and **deploying** your application, and provides building blocks for **creating CI/CD pipelines**. This enables you to focus on iterating on your application locally while Skaffold **continuously deploys** to your local or remote Kubernetes cluster.

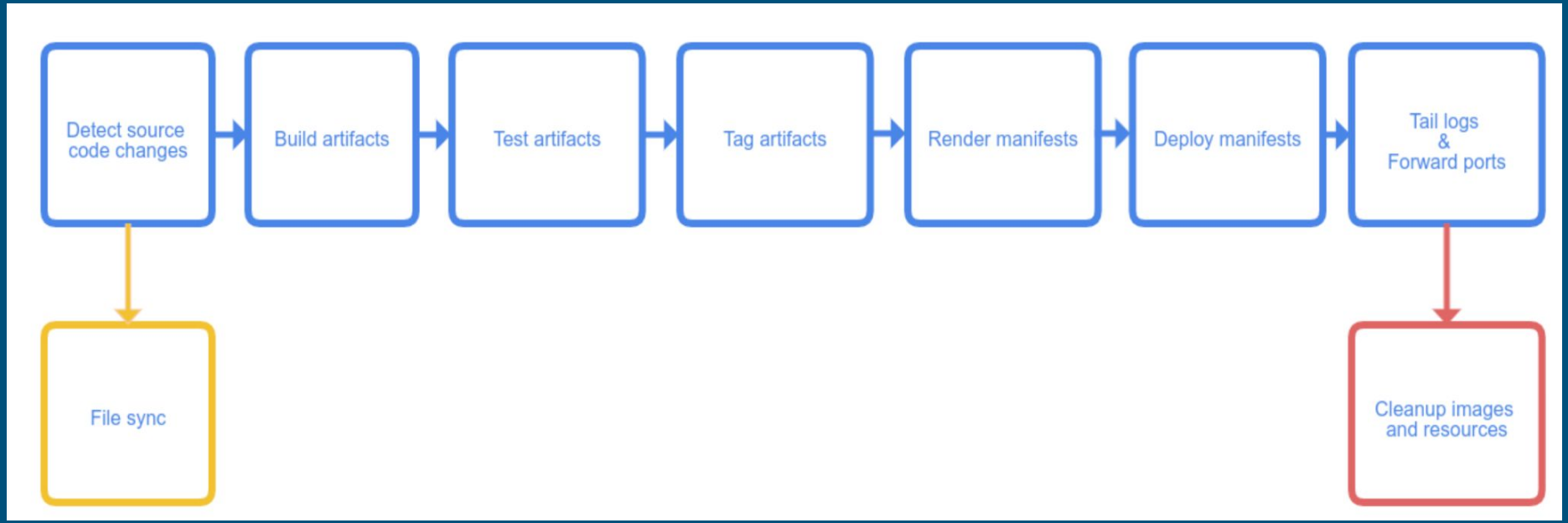
Pre-requisite

- Skaffold (version 1.37.0)
- Minikube (local kubernetes cluster)
- docker
- Kubectl
- Helm
- Java (1.8)
- maven

Skaffold framework

- Code language: golang
- cobra.Command
  - Main function (https://github.com/GoogleContainerTools/skaffold/blob/089a01b57a8be70d9c634363caa78af7c381bf77/cmd/skaffold/skaffold.go)
    - Root cmd (https://github.com/GoogleContainerTools/skaffold/blob/089a01b57a8be70d9c634363caa78af7c381bf77/cmd/skaffold/app/skaffold.go)
      - Sub cmd (https://github.com/GoogleContainerTools/skaffold/tree/089a01b57a8be70d9c634363caa78af7c381bf77/cmd/skaffold/app/cmd)
        - build
        - Deploy
        - Test
        - .....

Workflow


Detect source code changes
Build artifacts
Test artifacts
Tag artifacts
Render manifests
Deploy manifests
Tail logs & Forward ports
File sync
Cleanup images and resources

Core feature

- Fast local Kubernetes Development
  - optimized “Source to Kubernetes”
  - continuous feedback
- Skaffold projects work everywhere
  - share with other developers (**git clone & skaffold run**)
  - CI/CD building blocks (**skaffold build**, **skaffold test** and **skaffold deploy**)
  - GitOps integration (**skaffold render**)
- Lightweight
  - client-side only
  - minimal pipeline

How to use

Use `brew install skaffold` to install skafflod

Use skaffold version to check it installs successfully.

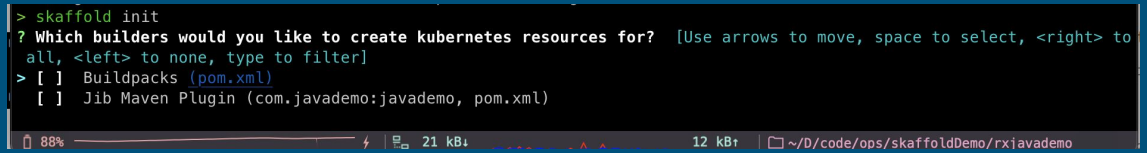
How to use – Skaffold init

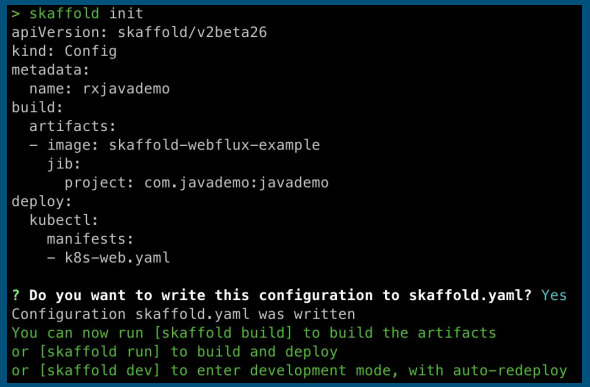
构建方式包含以下几种方式

1. Dockerfile
2. Jib
3. Buildpacks

PS: 当前项目中需要有k8s部署的manifest，如果没有的话，带上参数

`--generate-manifests`可以生成简单的k8s需要的manifest文件





How to use – skaffold build

- Local build
  - 使用本地的context来执行构建任务
  - 本地可以使用Docker，Maven，Gradle
- Cluster build（build完之后，会往docker registry推送镜像）
  - 使用Dockerfile
  - 使用Kaniko
-
-

As described above, the custom build script is expected to:

1. Build and tag the `$IMAGE` image
2. Push the image if `$PUSH_IMAGE=true`

Once the build script has finished executing, Skaffold will try to obtain the digest of the newly built image from a remote registry (if `$PUSH_IMAGE=true`) or the local daemon (if `$PUSH_IMAGE=false`). If Skaffold fails to obtain the digest, it will error out.

How to use – skaffold deploy

Supported deployers

- Kubectl
  - skaffold deploy --images skaffold-webflux-example
  - kubectl port-forward web-8c77ffbcb-95p4n 8080:8080
  - skaffold delete (删除应用)
- Helm
  - Need to install helm in your local machine
  - Helm create <chart name>
  - Add helm release into skaffold
  - skaffold deploy --images skaffold-webflux-example
- kustomize
- docker (does not deploy to Kubernetes: see documentation for more details)

How to use – skaffold test

- Custom Test
  - Enables users to run custom commands in the testing phase of the Skaffold pipeline
    - Unit test by shell
- Container Structure Test
  - brew install container-structure-test
    - Issues
      - https://github.com/GoogleContainerTools/skaffold/issues/3543
  - Enables users to validate built container images before deploying them to our cluster
  - Bash into docker container:
    - docker run --rm -it --entrypoint bash <your image name:tag>
  - Categories
    - Command Tests
    - File Existence Tests
    - File Content Tests
    - Metadata Test
  - Details: https://github.com/GoogleContainerTools/container-structure-test#command-tests

How to use – skaffold tag

- the gitCommit: tagger uses git commits/references.
  - is the default tag policy of Skaffold
- the inputDigest: tagger uses a digest of the artifact source files.
- the envTemplate: tagger uses environment variables.
- the datetime: tagger uses current date and time, with a configurable pattern.
- the customTemplate: tagger uses a combination of the existing taggers as components in a template.
- the sha256: tagger uses latest.

```
tagPolicy:
    envTemplate:
      template: "{{.FOO}}"
```

How to use – skaffold File Sync

- Hot deploy
- not rebuild and redeploy again just sync files
- Manual
  - manual sync rule must specify the src and dest field.
- Infer
  - Inferred sync mode only applies to modified and added files. File deletion will always cause a complete rebuild.
- Auto
  - Supported
    - Buildpacks (enable default)
    - Jib（https://github.com/GoogleContainerTools/skaffold/tree/main/examples/jib-sync）
- Side-effect: https://skaffold.dev/docs/pipeline-stages/filesync/#limitations

How to use – skaffold Log Tailing

- Log Tailing is enabled by default for **dev** and **debug**.
- Log Tailing is disabled by default for **run** mode; it can be enabled with the --tail flag.
- JSON Parsing
  - Filter specific log fields

How to use – skaffold Deploy Status Checking

- Related status: skaffold dev, ~ build, ~ debug, ~ deploy (enable default)
- Check points
  - Pod
  - Deployment
  - Stateful set
- Status check default time is 10min
- skaffold deploy --images skaffold-webflux-example --status-check

How to use – skaffold port farward

- Skaffold has built-in support for forwarding ports from exposed Kubernetes resources on your cluster to your local machine when running in dev, debug, deploy, or run modes.

```
portForward:
- resourceType: deployment
  resourceName: myDep
  namespace: mynamespace
  port: 8080
  localPort: 9000 # *Optional*
```

Using scenarios

- Local k8s deployment for running a application
- Google cloud deployment for remote CI/CD

Conclusion

- Advantages
  - Simple deploy locally
  - Support CI/CD
  - Open–resource tool
  - Google squad
  - Low learning cost

- Shortcomings
  - Some unknow issues
  - Non-mainstream tool

Feedback